

# 行政院國家科學委員會專題研究計劃成果報告 分散式系統開發學習環境

(Distributed System Development and study Environment on Java and CORBA)

計劃編號：NSC 89-2213-E-032-006

執行期限：88 年 8 月 1 日至 89 年 7 月 31 日

主持人：陳伯榮 私立淡江大學資訊工程系

## 一、中文摘要

眾所皆知地，並行、同步，以及不確定性將使得分散式系統的行為十分複雜。我們以三層的架構實作了一個分散式系統開發學習環境，簡稱為 DSDSE。

系統的第一層是提供一個瀏覽器的使用者介面，而由瀏覽器所呈現的隱藏式物件則是 Java 小程式(applets)。客戶端的小程式透過一個 IIOP ORB 來啟動在中間層的 CORBA 服務端物件。而服務端物件則藉著存取在最後一層的資料來提供所需的應用。

DSDSE 提供了讓使用者學習分散式系統理論及應用的環境。在此環境中，使用者可以藉著觀察分散式系統的相對應事件順序圖 (H-DAG) 及狀態轉移圖 (E-DAG) 來作交談式的學習。對任何可能的系統執行狀態變化的過程，此環境都可以提供視覺上動態的觀察，所以使用者可以看到他們所選擇的事件執行過程。隨機過程派翠網路也被採用來對分散式系統作定性分析(quality analysis)和定量分析(quantity analysis)。

關鍵詞：三層(3-Tier)系統，CORBA，Java，IIOP ORB，事件順序圖 (H-DAG)，狀態轉移圖 (E-DAG)，隨機過程派翠網路(STPN)。

## Abstract

It is well known that concurrency, synchronization, and nondeterminism make behavior of distributed systems very complicate. We use 3-Tier structure to implement a Distributed Systems Development and Study Environment (DSDSE).

The first tier is the user interface provided by a browser. The embedded components displayed by the browser are Java applets. Client applets invoke operations on CORBA middle-tier server objects via an IIOP ORB. The server objects provide the application logic and access the data in the final tier.

DSDSE presents the theory and practice of concurrent and distributed systems. In the environment, users can interactively study the distributed systems by observing the corresponding H-DAG and E-DAG. The visualization takes as input any possible execution trace of the respective system, so that user can view it in any possible execution that they may select. Also Stochastic Time Petri Nets (STPN) is adopted as the main model to analyze systems both qualitatively and quantitatively.

Keywords : 3-Tier system, CORBA, Java, IIOP ORB, H-DAG, E-DAG, STPN

## 二、緣由與目的

由於並行性，同步和不確定性，導致分散式系統的行為非常複雜，在學習和研究分散式系統的過程中，我們發現在求得理論和應用的平衡時，如果不瞭解應用就很難有一個良好的學習理論的動機。

本計畫的主要目標即是提供一個理想的分散式系統學習環境，進而透過此環境設計、開發分散式系統。我們提出了一個 3-Tier 架構的系統，稱為分散式系統開發學習環境，簡稱為 DSDSE。

我們以 CORBA[1]提供一個“分散式物件計算環境”做為整個系統的基礎；使用爪哇程式語言結合瀏覽器達成與使用者互動及動畫顯示的介面；用爪哇資料庫連結程式處理資料庫的存取；並以統一化模式語言

(Unified Modeling Language, 簡稱為 UML) [2]建構系統的藍圖，完成一個三層式(Three Tier)的分散式系統開發學習環境(如圖一)。

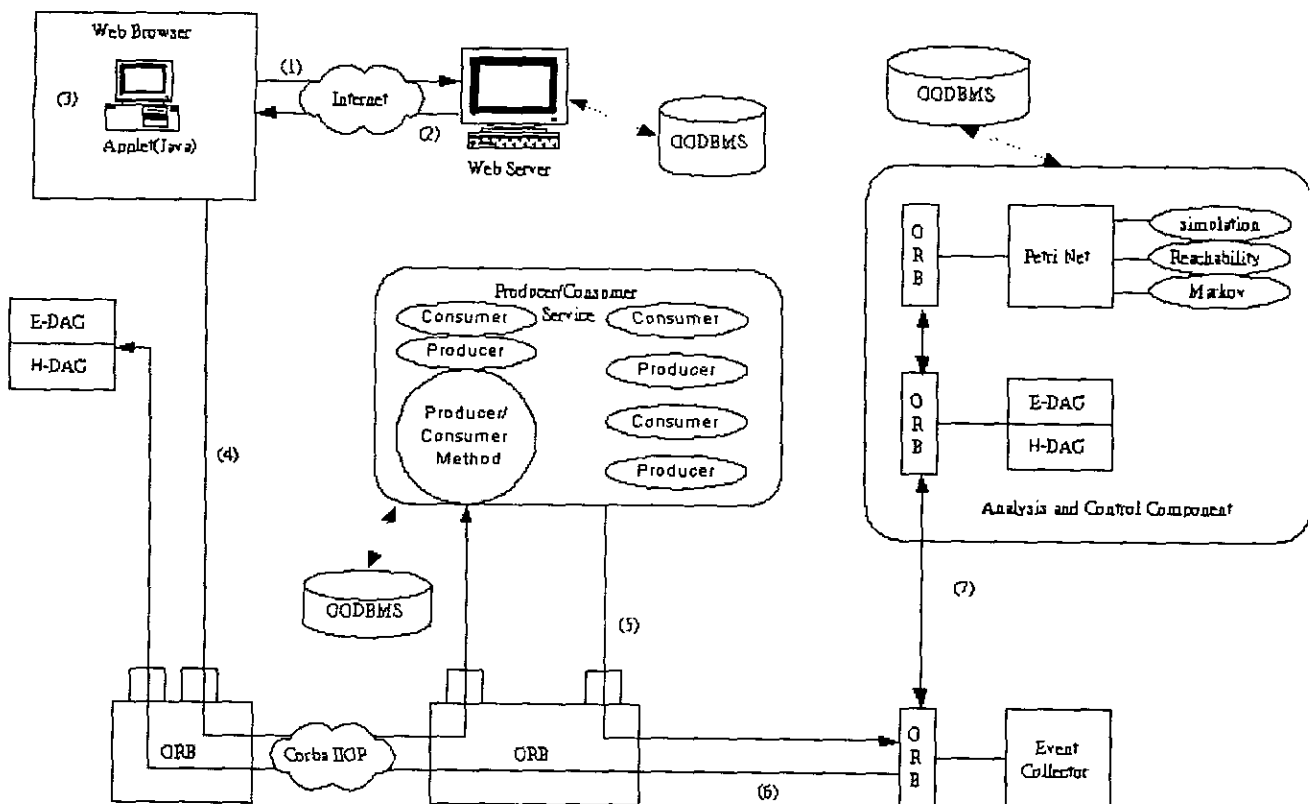
## 三、結果與討論

DSDSE 有兩個子系統，一個為互動式的分散式演算法執行環境子系統：其將分散式演算法執行於分散式物件計算環境中，將過程紀錄並顯示在客戶端的瀏覽器。另一則為互動式的分散式演算法分析環境子系統：主要利用隨機過程派翠網路[3]完成 DSDSE 中之演算法分析。

我們分別依系統設計與實作討論：

在系統設計方面，本計劃完成 DSDSE 二個主要的 Use Case；其中演算法演繹 Use Case 由演算法執行環境子系統負責完成。可以歸納出以下四種主要的功能：

- (1) 系統允許使用者設定參數，並有 Default 設定值，參數的種類有使用者角色、使用者人數、執行方式三種；
- (2) 系統以動畫的方式呈現演算法運作的過程及使用者參與的狀態。
- (3) 執行演算法於分散式環境中，依演算法的特性，系統可允許使用者中途離開或加入。



圖一：DSDSE 三層架構圖

(4) 系統可以重演前次執行的過程。

這個子系統中幾個主要的類別為：

- (1) 演算法管理者
- (2) 分散式系統
- (3) 動畫系統
- (4) 事件順序圖(H-DAG)
- (5) 工作者管理員
- (6) 演算法物件
- (7) 演算法資料庫系統
- (8) 資料記錄

演算法分析 Use Case 由分散式演算法分析環境子系統負責完成。這個子系統分為五個部份：解 P-永不變特性、可到達樹建構、分析解、PN 互動式模擬以及派翠網路自動模擬。詳細設計過程與相關藍圖(UML Diagram)請參照[4][5]。

在實作方面：

我們以 CORBA 的 POA[6] (Portable Object Adapter) 為中心建立一個工作者管理員，由工作者管理員控制 CORBA 物件的起始及結束。在這種架構中，每個以 CORBA 物件完成的演算法參與者，都獨立的執行於網路上的某一個電腦上，組成一個執行分散式演算法的群組，群組中有一個特殊的 CORBA 物件，稱為演算法管理員，負責收集資料，及與使用者溝通。

以最常見的 Count 範例，說明以 CORBA POA 撰寫 Three Tier 程式的方式。工作環境為 WIN NT 4.0，JDK 1.2.2 及 JacORB 1.0[7]。

首先實作 CORBA 物件

(1) 以 IDL 描述 CORBA 物件

Count 物件的 IDL 範例如下：

```
module CountApp {  
    interface Count {  
        attribute long sum;  
        void increment();  
    };  
};
```

(2) IDL Compiler

以 Count 物件為例說明 IDL Compiler 所產

生的主要檔案為：CountPOA、CountStub、Count、CountOperations、CountHelper。

(3) 撰寫物件實體—Servant

將 IDL 檔編譯之後，接下來必須完成 CORBA 物件。由於 CORBA IDL 已經將在分散式網路上的一些繁瑣的問題都隱藏起來，所以我們要作的僅是完成物件的介面，也就是完成 CORBA 物件提供服務實體—Servant。

在撰寫物件必須繼承由 IDL Compiler 所產生的 POA Class，以本例來說必須繼承 CountPOA 這個 Class，完成 increment() 這個介面。

再來完成 CORBA 物件的 Server 程式：

CORBA 物件已經完成，接下來就是撰寫物件服務端及使用者介面的程式。

撰寫 CORBA 物件 Server 端程式的步驟說明如下。

(1) initialize ORB：

使用 ORB.init( args, null )

(2) 獲得 Root POA：

使用 orb.resolve\_initial\_references  
("RootPOA")

(3) 新建 POA Policy：

用 policies = new org.omg.CORBA.Policy  
建立 Policy 物件。

(4) 建立新的 POA：

用 poa

=rootPOA.create\_POA( "persistent",  
poa\_mgr, policies ) 建立新的 POA。  
CORBA

POA 必須新增在 Root POA 之下。

(5) 建立 CORBA 物件：

CountImpl servant = new CountImpl()

(6) 啟動物件並向 POA 登記(register)：

poa.activate\_object\_with\_id( oid, servant )

(7) 取得名稱伺服器(Name Service)的參考：

objRef = orb.resolve\_initial\_references  
("NameService")

(8) 將物件向名稱伺服器登記：

ncRef.bind( path, countobj )

(9) 讓 ORB 進入等待使用者介面提出需求的

狀態：用 orb.run()

Three Tier 軟體架構中 CORBA 物件的 Server 端必須有能力存取資料庫。實作 CORBA 物件的 Server 端時，使用 JDBC 的方式與一般 JAVA 程式設計方式相同，這裡不多作介紹。

最後撰寫 CORBA 物件的 Client 程式以 CORBA 撰寫 Three Tier 架構的程式，使用者介面必須能透過 ORB 呼叫 CORBA 物件，使用者介面撰寫的原則是一取得服務端物件的參考，用物件參考直接呼叫服務端提供服務。

以 Count 物件為範例。說明如下：

initialize ORB：使用 ORB.init( args, null )

取得名稱伺服器的參考：

```
objRef = orb.resolve_initial_references  
("NameService")
```

(3) 由名稱伺服器取得物件的參考：

```
NameComponent nc = new
```

```
NameComponent("POACount","");
```

```
NameComponent path[] = {nc};
```

```
Count stubCount = CountHelper.narrow  
( ncRef.resolve(path) );
```

(4)透過物件參考直接對物件送出服務需求。如：stubCount.increment();

以 CORBA 與 Java 實做的 Three Tier 軟體架構中，Java Applet 提供使用者介面的網頁向 CORBA 物件提出服務需求的能力。

#### 四、計劃成果自評

我們用 CORBA 與 JAVA 完成了一個能接受使用者設定，在分散式環境下執行與分析分散式演算法的互動式環境，用以支援 DSDSE 系統完成教學的功能。並以物件導向的觀念利用 UML Diagram 分析並設計系統。這個環境使用獨立的 CORBA 物件作為執行分散式演算法的參與者，這種架構也可以擴充用來製作一個分散式運算的模擬器，或者是提供一個平行運算的環境。

未來此系統尚有一些功能可以加入：

(1)在教學完成後做測試，讓使用者知道自

己了解多少。

(2)允許多人網路連線，共同模擬一個演算法。

(3)在多人環境下，允許使用者中途離開或加入。

(4)者在多人環境下，可以看見有幾個 User 在一起運作以及各使用者所擔任的腳色。

(5)使用者可以線上交談或向在線上的老師發問

#### 五、參考文獻

[1]The Common Object Request Broker: Architecture and Specification, <http://www.omg.org/library/whitepapers.html>, Revision 2.3, June 1999.

[2][Booch 1999]

Grady Booch, James Rumbaugh, and Ivar Jacobson, "The Unified Modeling Language User Guide," Addison Wesley Longman, 1999.

[3][Ajmone 1990]

M. Ajmone Marsan, G. Chiola, "Stochastic Petri Nets: An Elementary Introduction", in: G. Rozenberg (editor), Advances in Petri Nets 1989, Lecture Notes on Computer Science, n. 424, Springer Verlag, 1990.

[4][王志勇 2000]

王志勇, "分散式系統開發學習環境之演算法執行環境子系統", 私立淡江大學資訊工程研究所碩士論文, 2000, 6.

[5][葉金川 2000]

葉金川, "分散式系統開發學習環境之演算法分析子系統", 私立淡江大學資訊工程研究所碩士論文, 2000, 7.

[6]Object Management Group, Specification of the Portable Object Adapter(POA), OMG Document orbos/97-05-15 ed., June 1997

[7]JacORB Programming Guide, v1.0, Gerald Brose, Institut Fur Informatik Freie Universitat Berlin, Germany, [Brose@inf.fu-berlin.de](mailto:Brose@inf.fu-berlin.de), December, 2, 1998.